```
// This source code is subject to the terms of the Mozilla Public License 2.0 at https://mozilla.org/M
// Copyright (c) 2023 trademasterindicator. All rights reserved.
// The Pinescript source code ("MACD Bands - Multi Timeframe [TradeMaster Lite]") is exclusively licen
// By accessing or using the script, you agree to the following terms:
// Grant of License: You are granted a non-transferable license to use the script for personal, non-cc
// Ownership and Intellectual Property: trademasterindicator retains all ownership and intellectual pr
// Disclaimer of Warranty: The source code is provided as-is, without warranties. trademasterindicator
// Termination: This license is valid until terminated.
//@version=5

indicator("MACD Bands - Multi Timeframe [TradeMaster Lite]", 'MACD Bands - MTF [TradeMaster Lite]', pr


// INPUTS \\
i_src    = input.source   (close, 'Source'   , inline = 'src')
i_tf     = input.timeframe(''   , 'Timeframe', inline = 'src', tooltip = 'Only timeframes higher than
i_maType = input.string   ('EMA', 'MA Type'  , inline = 'ma' , options = ['SMA','SMMA','EMA','DEMA','T
i_bbMult = input.float    (1    , 'Band mult', inline = 'ma' , minval = 0.1, maxval = 100, step = 0.1)
i_fast   = input.int      (12   , ''         , inline = 'len')
i_slow   = input.int      (26   , ''         , inline = 'len')
i_sign   = input.int      (9    , ''         , inline = 'len')


// OBJECT BLUEPRINT \\
type Macd
    float line = na
    float sign = na
    float hist = na
    float top  = na
    float bot  = na


// FUNCTIONS \\
get_sec(tf , exp) => request.security('', tf, exp[1], barmerge.gaps_off, barmerge.lookahead_on)
dema   (src, len) => ema = ta.ema(src, len), 2 * ema - ta.ema(ema,len)
tema   (src, len) => ema = ta.ema(src, len), ema2 = ta.ema(ema, len), 3 * (ema - ema2) + ta.ema(ema2,

get_ma(src, len) =>
    switch i_maType
        'SMA'  => ta.sma   (src, len)
        'SMMA' => ta.rma   (src, len)
        'EMA'  => ta.ema   (src, len)
        'DEMA' => dema     (src, len)
        'TEMA' => tema     (src, len)
        'LSMA' => ta.linreg(src, len, 0)
        'HMA'  => ta.hma   (src, len)
        'VWMA' => ta.vwma  (src, len)
        'WMA'  => ta.wma   (src, len)

get_macd() =>
    macd      = get_ma  (i_src, i_fast) - get_ma(i_src, i_slow)
    sign      = get_ma  (macd , i_sign)

```

```
    deviation = ta.stdev(sign , i_sign) * i_bbMult
    Macd.new(macd, sign, macd - sign, sign + deviation, sign - deviation)


// CALCULATION \\ exclued ltf for now, irrelevant and noisy anyways
var color clrHist = na
var isMtf = not na(i_tf) and timeframe.in_seconds(i_tf) > timeframe.in_seconds()
tf_change = isMtf ? timeframe.change(i_tf) : true
macd      = get_macd()
macd     := isMtf ? get_sec(i_tf, macd) : macd
clrHist  := macd.hist > macd.hist[1] ? #00897b83 : macd.hist < macd.hist[1] ? #ff990080 : clrHist


// PLOTS \\
isBull   =  macd.line > macd.sign
macdline = plot(tf_change ? macd.line : na, 'macdLine', isBull ? #4caf50 : #ff5252)
plot(tf_change ? macd.sign : na, 'signalLine', #ffeb3b)

plot(macd.hist, 'histogram ', clrHist, style = plot.style_columns)
plot(ta.cross(macd.line, macd.sign) ? macd.line : na, 'cross', isBull ? #00897b : #ff9800, 4, plot.sty

bTop = plot(tf_change ? macd.top : na, 'band top', #2196f31a)
bBot = plot(tf_change ? macd.bot : na, 'band bot', #2196f31a)


// FILLS \\
fill(bTop    , bBot,                         #2196f31a     , 'Fill band', fillgaps = true)
fill(macdline, bTop, macd.line > macd.top ? #4caf4f7d : na, 'Fill bull', fillgaps = true)
fill(macdline, bBot, macd.line < macd.bot ? #ff525288 : na, 'Fill bear', fillgaps = true)

// a zero line :) \\
hline(0, '0 Line', linestyle = hline.style_dotted, color = color.gray, editable = false)
```
